Listing Program

```
// Definisi Pin
const int knockSensor = 0;          // Piezo sensor pada pin0
const int programSwitch = 2;        // Button pada pin 2
const int solenoid = 3;            // Solenoid pada pin 3
const int redLED = 4;               // Status LED
const int greenLED = 5;             // Status LED


const int threshold = 50;           // Ambang batas sinyal
minimum yang terbaca sebagai ketukan
const int rejectValue = 25;         // Batas toleransi
const int averageRejectValue = 15; // Batas rata-rata
toleransi
const int knockFadeTime = 150;      // Jarak waktu sebelum
membaca ketukan selanjutnya


const int maximumKnocks = 21;       // Batas maksimum
ketukan
const int knockComplete = 1200;     // Waktu tunggu
menganggap telah selesai mengetuk



// Variabel.
int secretCode[maximumKnocks] = {100, 100, 100, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
int lockCode[maximumKnocks] = {100, 100, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
int knockReadings[maximumKnocks];   // Ketika di ketuk array
ini akan terisi hingga batas maksimum ketukan
int knockSensorValue = 0;           // Reset nilai sensor
int programButtonPressed = false;   // menetapkan posisi
pengubah program ketika tidak ditekan adalah false


void setup() {
```

```
  pinMode(solenoid, OUTPUT);
  pinMode(redLED, OUTPUT);
  pinMode(greenLED, OUTPUT);
  pinMode(programSwitch, INPUT);

  Serial.begin(9600);
  Serial.println("Program start.");

  digitalWrite(greenLED, HIGH);
}

void loop() {
  // Listen for any knock at all.
  knockSensorValue = analogRead(knockSensor);

  if (digitalRead(programSwitch)==HIGH){  // button bernilai
HIGH
    programButtonPressed = true;          // karena button
ditekan posisi pengubah program menjadi true
    digitalWrite(redLED, HIGH);
  } else {
    programButtonPressed = false;
    digitalWrite(redLED, LOW);
  }

  if (knockSensorValue >=threshold){
    listenToSecretKnock();
  }
}

// Records the timing of knocks.
void listenToSecretKnock(){
```

```
  Serial.println("ketukan mulai");


  int i = 0;
  // First lets reset the listening array.
  for (i=0;i<maximumKnocks;i++){
    knockReadings[i]=0;
  }


  int currentKnockNumber=0;                        // Inkremen
array
  int startTime=millis();                          // titik
awal
  int now=millis();


  digitalWrite(greenLED, LOW);
  if (programButtonPressed==true){
     digitalWrite(redLED, LOW);
  }
  delay(knockFadeTime);
  digitalWrite(greenLED, HIGH);
  if (programButtonPressed==true){
     digitalWrite(redLED, HIGH);
  }
  do {
    //listen for the next knock or wait for it to timeout.
    knockSensorValue = analogRead(knockSensor);
    if (knockSensorValue >=threshold){                    //
jika nilai melebihi Ambang batas
      //record the delay time.
      Serial.println("ketuk.");
      now=millis();
      knockReadings[currentKnockNumber] = now-startTime;
```

```
      currentKnockNumber ++;
//inkremen ketukan
      startTime=now;
      // and reset our timer for the next knock
      digitalWrite(greenLED, LOW);
      if (programButtonPressed==true){
        digitalWrite(redLED, LOW);
      }
      delay(knockFadeTime);
      digitalWrite(greenLED, HIGH);
      if (programButtonPressed==true){
        digitalWrite(redLED, HIGH);
      }
    }

    now=millis();

    //did we timeout or run out of knocks?
  } while ((now-startTime < knockComplete) &&
(currentKnockNumber < maximumKnocks));

  //we've got our knock recorded, lets see if it's valid
  if (programButtonPressed==false){             // Jika
button tidak ditekan
    if (validateKnock() == true){              // dan ketukan
buka benar triggerDoorUnlock
      triggerDoorUnlock();
    } else {
      digitalWrite(greenLED, LOW);
      for (i=0;i<4;i++){
        digitalWrite(redLED, HIGH);
        delay(100);
        digitalWrite(redLED, LOW);
```

```
        delay(100);
      }
    }
    if (invalidateKnock() == true){       // jika ketukan
tutup benar triggerDoorLock
      triggerDoorlock();
    }
  } else { // jika ketukan benar tapi masih dalam mode ubah
program tidak dilakukan apa-apa
    validateKnock();
    Serial.println("Ketukan Baru Disimpan.");
    Serial.println(" ");
    digitalWrite(redLED, LOW);
    digitalWrite(greenLED, HIGH);
    for (i=0;i<3;i++){
      delay(100);
      digitalWrite(redLED, HIGH);
      digitalWrite(greenLED, LOW);
      delay(100);
      digitalWrite(redLED, LOW);
      digitalWrite(greenLED, HIGH);
    }
  }
}



// Aktifkan Relay
void triggerDoorUnlock(){
  Serial.println("Kunci Terbuka");
  Serial.println(" ");
  int i=0;
```

```
  digitalWrite(solenoid, HIGH  );
  digitalWrite(greenLED, HIGH);

  for (i=0; i < 5; i++){
      digitalWrite(greenLED, LOW);
      delay(100);
      digitalWrite(greenLED, HIGH);
      delay(100);
  }

}

// Nonaktifkan relay
void triggerDoorlock(){
  Serial.println("Pintu Terkunci");
  Serial.println(" ");
  int i=0;

.
  digitalWrite(solenoid, LOW);
  digitalWrite(greenLED, LOW);


  for (i=0; i < 5; i++){
      digitalWrite(greenLED, LOW);
      delay(100);
      digitalWrite(greenLED, HIGH);
      delay(100);
  }
```

```
}
\
boolean validateKnock(){
  int i=0;

  int currentKnockCount = 0;
  int secretKnockCount = 0;
  int maxKnockInterval = 0;

  for (i=0;i<maximumKnocks;i++){
    if (knockReadings[i] > 0){
      currentKnockCount++;
    }
    if (secretCode[i] > 0){                                //
inkremen kode ketukan
      secretKnockCount++;
    }

    if (knockReadings[i] > maxKnockInterval){
      maxKnockInterval = knockReadings[i];
    }
  }

  if (programButtonPressed==true){
      for (i=0;i<maximumKnocks;i++){ // normalize the times
        secretCode[i]= map(knockReadings[i],0,
maxKnockInterval, 0, 100);
      }
      digitalWrite(greenLED, LOW);
      digitalWrite(redLED, LOW);
      delay(1000);
      digitalWrite(greenLED, HIGH);
```

```
      digitalWrite(redLED, HIGH);
      delay(50);
      for (i = 0; i < maximumKnocks ; i++){
        digitalWrite(greenLED, LOW);
        digitalWrite(redLED, LOW);
        // only turn it on if there's a delay
        if (secretCode[i] > 0){
          delay( map(secretCode[i],0, 100, 0,
maxKnockInterval)); // Expand the time back out to what it
was.  Roughly.
          digitalWrite(greenLED, HIGH);
          digitalWrite(redLED, HIGH);
        }
        delay(50);
      }
       return false;
  }


  if (currentKnockCount != secretKnockCount){
    return false;
  }


  //  Penghitungan Interval
   int totaltimeDifferences=0;
  int timeDiff=0;
  for (i=0;i<maximumKnocks;i++){
    knockReadings[i]= map(knockReadings[i],0,
maxKnockInterval, 0, 100);
    timeDiff = abs(knockReadings[i]-secretCode[i]);
    if (timeDiff > rejectValue){
      return false;
    }
    totaltimeDifferences += timeDiff;
```

```
  }
  if
(totaltimeDifferences/secretKnockCount>averageRejectValue){
    return false;
  }


 return true;
}
boolean invalidateKnock(){
  int i=0;


  // simplest check first: Did we get the right number of
knocks?
  int currentKnockCount = 0;
  int secretKnockCount = 0;
  int maxKnockInterval = 0;


  for (i=0;i<maximumKnocks;i++){
    if (knockReadings[i] > 0){
      currentKnockCount++;
    }
    if (lockCode[i] > 0){
      secretKnockCount++;
    }


    if (knockReadings[i] > maxKnockInterval){
      maxKnockInterval = knockReadings[i];
    }
  }
  if (currentKnockCount != secretKnockCount){
    return false;
  }
```

```
  //  Penghitungan interval
  int totaltimeDifferences=0;
  int timeDiff=0;
  for (i=0;i<maximumKnocks;i++){ // Normalize the times
    knockReadings[i]= map(knockReadings[i],0,
maxKnockInterval, 0, 100);
    timeDiff = abs(knockReadings[i]-lockCode[i]);
    if (timeDiff > rejectValue){
      return false;
    }
    totaltimeDifferences += timeDiff;
  }
  if
(totaltimeDifferences/secretKnockCount>averageRejectValue){
    return false;
  }

  return true;

}
```